involves the traversal of links between data structure element 245 as is well known in the art.

[0023] The state transition from the pre-associated state to the pending insert state similarly involves navigation to the point of insertion in the data structure 240. The state transition from the pending insert state to the hidden state also involves navigation to the insertion point.

[0024] During the update phase of step 130 (FIG. 1), update data 250 is developed by computer code 220 (FIG. 2) in the form of a First In Last Out (FILO) stack. Update data 250 includes address data relating to the locations of elements 245. This address data is used in the commit phase of step 140 (FIG. 1) in place of navigating data structure 240 (FIG. 2). Update data 250 further includes instructions for inserting and deleting data structure elements as further described herein.

[0025] FIG. 3 is a block diagram illustrating a relationship between update data 250 (FIG. 2) and a row identifier index (RID) 310 of a type well known in the art. The RID 310 is used to provide a convenient means for accessing update data 250. RID 310 is a table of pointers 315 pointing at table rows 320 that hold elements 245 of data structure 240 (FIG. 2).

[0026] In the present invention each table row 320 includes a pending update field (PU) 330 that holds a pointer to update data 250 if update data 250 associated with that row 320 exists.

If a specific PU field 330 holds a NULL pointer created when a specific row 320 is developed, then update data 250 associated with the specific row 320 does not exist and no updates are pending on a particular element 245 of data structure 240 associated with the specific row 320. If PU 330 holds a non-NULL pointer then the pointer points to update data 250 associated with the particular element 245.

[0027] Update data 250 is developed during the update phase of step 130 (FIG. 1). For each task in task queue 260 (FIG.2) that includes a commit phase state transition, a new top item 340 is added to update data 250. The previous top item becomes a second item (not shown) in the FILO stack of update data 250. Top item 340 includes information to be used in the commit phase of step 140 (FIG.1) to complete the operation. For example, the information includes instructions for changing the state of element 245 from the pending insert state to the valid state.

[0028] When added to update data 250, top item 340 includes a flag (not shown) indicating that conflicts have not been checked. Upon encountering such a flag, computer code 220 searches update data 250 for possible conflicts including operations executing upon the same element 245. If no conflicts are found then the flag is modified to indicate that no conflicts exist. If a conflict is found then a rollback of the

operation is scheduled for the next commit phase as further described herein, and in particular with reference to FIG. 5.

[0029] During the commit phase of step 140 (FIG. 1) computer code 220 traverses RID 310 and examines each row 320 for a non-NULL PU field 330. For each non-NULL PU field 330, update data 250 is processed from the top down. When the last operation instructions in the update data 250 are executed, the associated PU 330 is reset to NULL and computer code 220 proceeds to the next pointer 315 in the RID 310 table.

[0030] The instructions from update data 250 include a pointer to the data elements 245 within data structure 240 operated on during the update phase of step 130. These pointers eliminate the need for computer code 220 to navigate through data structure 240 during the commit phase of step 140. The FILO structure of update data 250 forces operations to be performed in an order that assures no conflicts.

[0031] An element insertion operation includes, as stated previously, the state transitions from pre-associated to pending insert to valid. FIG. 4A illustrates the result of a state transition from the pre-associated state to the pending insert state at the conclusion of the update phase of step 130 (FIG. 1). A new element 410 of a type well known in the art is being inserted into data structure 240 between a first element 420 and a second element 430. An existing link 440 points from